Implementing Efficient and Scalable Flow Control Schemes in MPI over InfiniBand

Jiuxing Liu and Dhabaleswar K. Panda

Computer Science and Engineering
The Ohio State University

Presentation Outline

- Introduction and overview
- Flow control design alternatives
- Performance Evaluation
- Conclusion

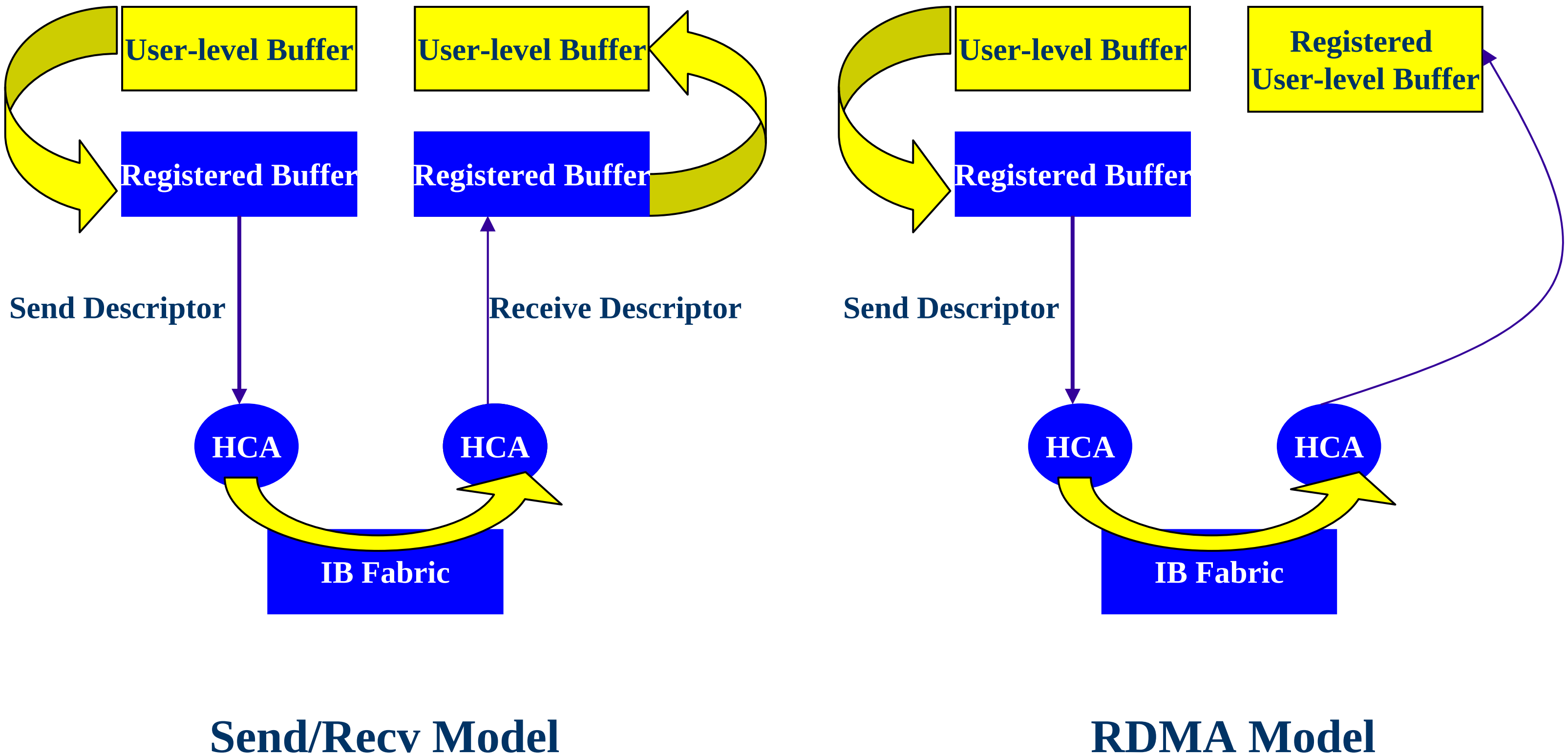
Introduction

- InfiniBand is becoming popular for high performance computing
- Flow control is an important issue in implementing MPI over InfiniBand
  - Performance
  - Scalability

InfiniBand Communication Model

- Different transport services
  - Focus on Reliable Connection (RC) in this paper
- Queue-pair based communication model
- Communication requests posted to send or receive queues
- Communication memory must be registered
- Completion detected through Completion Queue (CQ)

InfiniBand Send/Recv and RDMA Operations

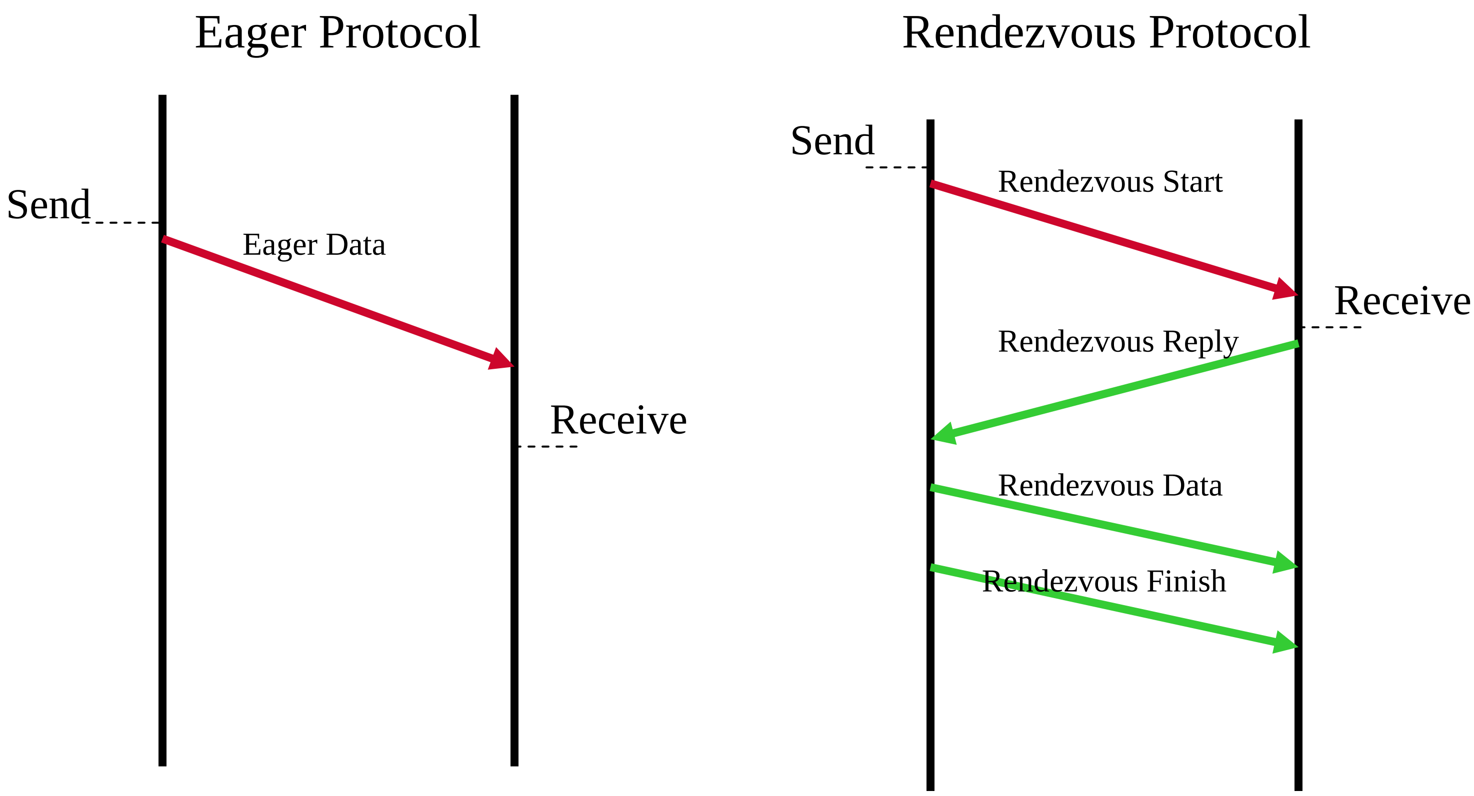


- For Send/Recv, a send operation must be matched with a pre-posted receive (which specifies a receive buffer)

End-to-End Flow Control Mechanism in InfiniBand

- Implemented at the hardware level
- When there is no recv buffer posted for an incoming send packet
  - Receiver sends RNR NAK
  - Sender retries

MPI Communication Protocols



Expected and Unexpected Messages in MPI Protocols

- Expected Messages:
  - Rendezvous Reply
  - Rendezvous Data
  - Rendezvous Finish
- Unexpected Message:
  - Eager Data
  - Rendezvous Start

Why Flow Control is Necessary

- Unexpected messages need resources (CPU time, buffer space, etc)
- MPI itself does not limit the number of unexpected messages
  - Receiver may not be able to keep up
  - Resources may not be enough
- Flow control (in the MPI implementation) is needed to avoid the above problems

InfiniBand Operations used for Protocol Messages

- Send/Recv operations used for Eager protocol and control messages in Rendezvous protocol
  - Can also exploit RDMA (not used in this paper)
- RDMA used for Rendezvous Data
- Unexpected messages are from Send/Recv

Presentation Outline

- Introduction and overview
- Flow control design alternatives
- Performance Evaluation
- Conclusion

Flow Control Design Outline

- Common issues
- Hardware based
- User-level static
- User-level dynamic

Flow Control Design Objectives

- Need to be effective
  - Preventing the receiver from being overwhelmed
- Need to be efficient
  - Very little run time overhead
  - No unnecessary stall of communication
  - Efficient buffer usage
    - How many buffers for each connection

Classification of Flow Control Schemes

- Hardware based vs. User-level
  - Hardware based schemes exploit InfiniBand end-to-end flow control
  - User-level schemes implements flow control in MPI implementation
- Static vs. Dynamic
  - Static schemes use a fixed number of buffers for each connection
  - Dynamic schemes can adjust the number of buffers during execution

Hardware-Based Flow Control

- No flow control in MPI
- Rely on InfiniBand end-to-end flow control
- Implemented entirely in hardware and transparent to MPI

Advantages and Disadvantages of the Hardware-Based Scheme

- Advantages
  - Almost no run-time overhead at the MPI layer during normal communication
  - Flow control mechanism makes progress independent of application
- Disadvantages
  - Very little flexibility
  - The hardware flow control scheme may not be the best for all communication patterns
  - Separation of buffer management and flow control
    - No information to MPI to adjust its behavior
    - Difficult to implement dynamic schemes

User-Level Static Schemes

- Flow control handled in MPI implementation
- Fixed number of buffers for each connection
- Credit-based scheme
  - Piggybacking
  - Explicit credit messages

Problems of the User-Level Static Scheme

- More overhead at the MPI layer (for credit management)
- Flow control progress depends on application
- Buffer usage is not optimal, may result in:
  - Wasted buffer for some connections
  - Unnecessary communication stall for other connections

User-Level Dynamic Schemes

- Similar to the user-level static schemes
- Start with only a few buffers for each connection
- Use a feedback based control mechanism to adjust the number of buffers based on communication pattern

User-Level Dynamic Scheme Design Issues

- How to provide information feedback
  - When no credit, sender will put a message into a "backlog"
  - Message will be sent when more credits are available
  - Tag messages to indicate if they have gone through the backlog
- How to respond to feedback
  - Increase the number of buffers for the connection if a receiver gets a message that has gone through the backlog
  - Linear or exponential increase

Presentation Outline

- Introduction and overview
- Flow control design alternatives
- Performance Evaluation
- Conclusion

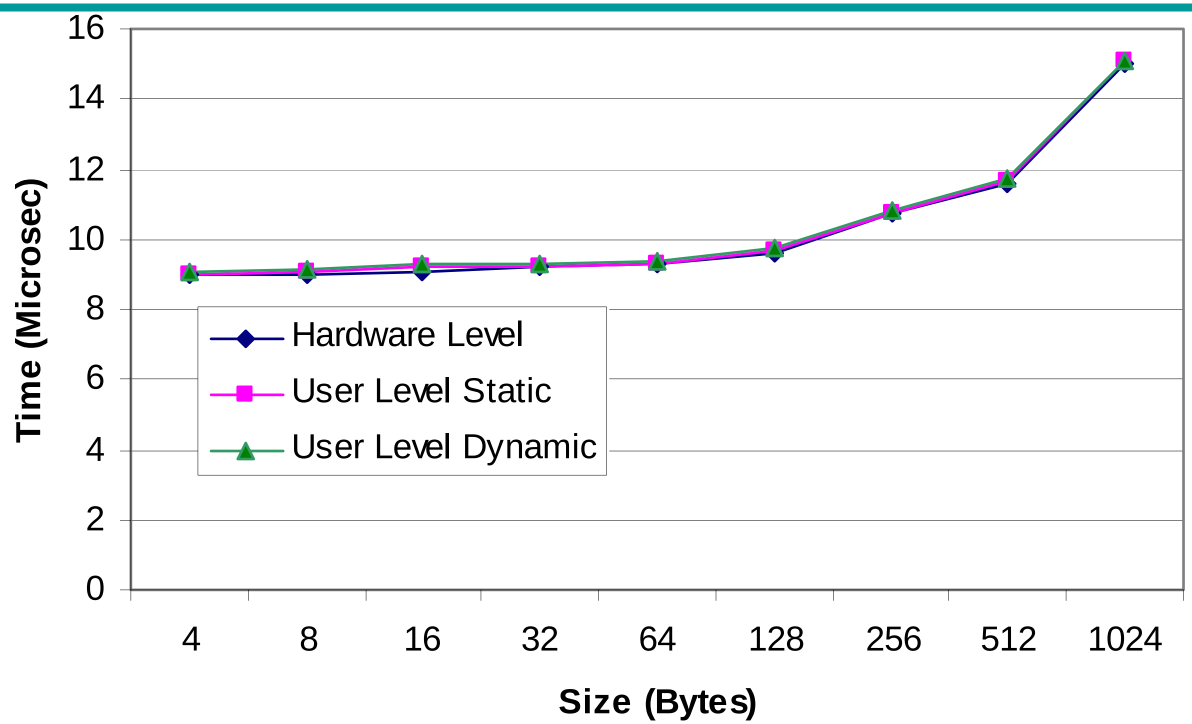
Experimental Testbed

- 8 SuperMicro SUPER P4DL6 nodes (2.4 GHz Xeon, 400MHz FSB, 512K L2 cache)
- Mellanox InfiniHost MT23108 4X HCAs (A1 silicon), PCI-X 66bit 133MHz
- Mellanox InfiniScale MT43132 switch

Outline of Experiments

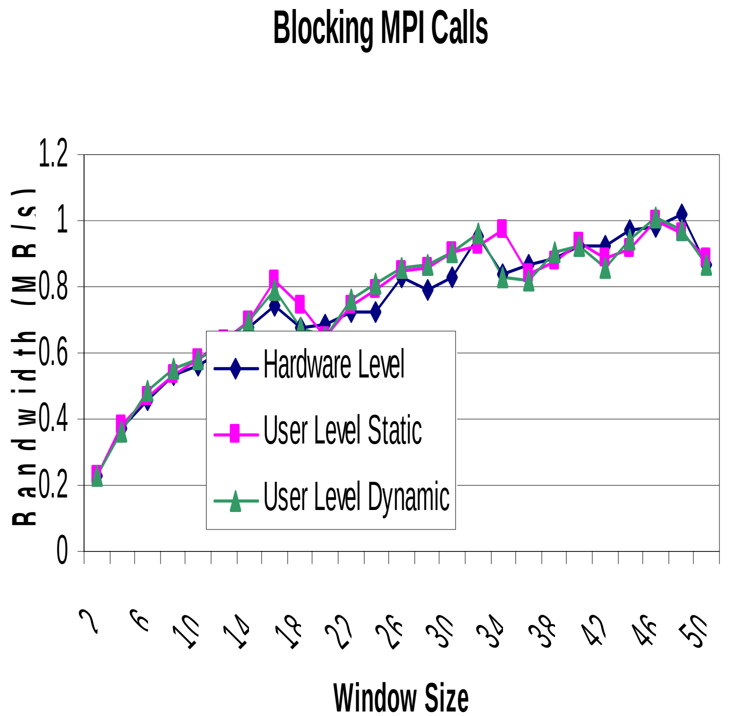
- Microbenchmarks
  - Latency
  - Bandwidth
    - MPI Blocking and Non-blocking Functions
    - Small and large messages
- NAS Benchmarks
  - Running time
  - Communication characteristics related to flow control

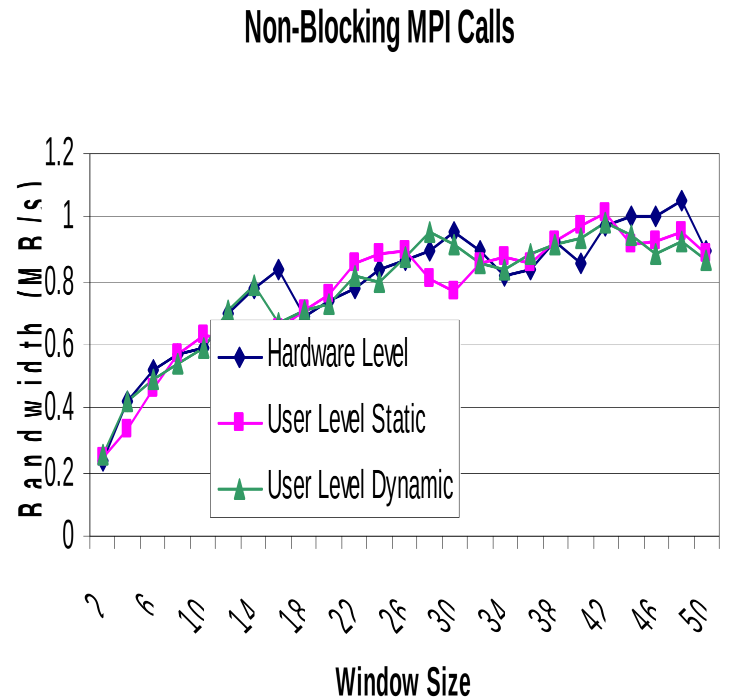
MPI Latency



- In latency tests, flow control usually is not an issue because communication is symmetric
- All schemes perform the same, which means that user level overhead is very small in this case

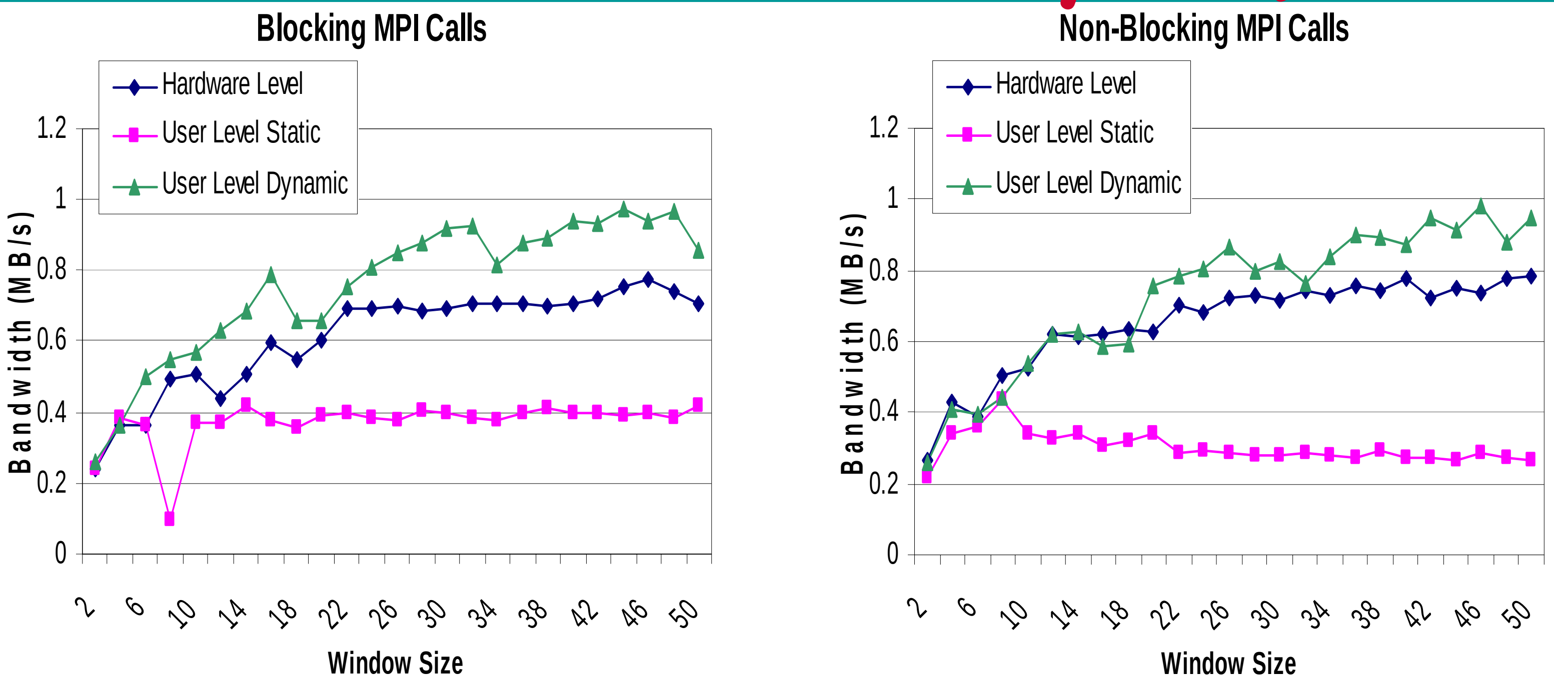
MPI Bandwidth (Prepost = 100 and Size = 4 Bytes)





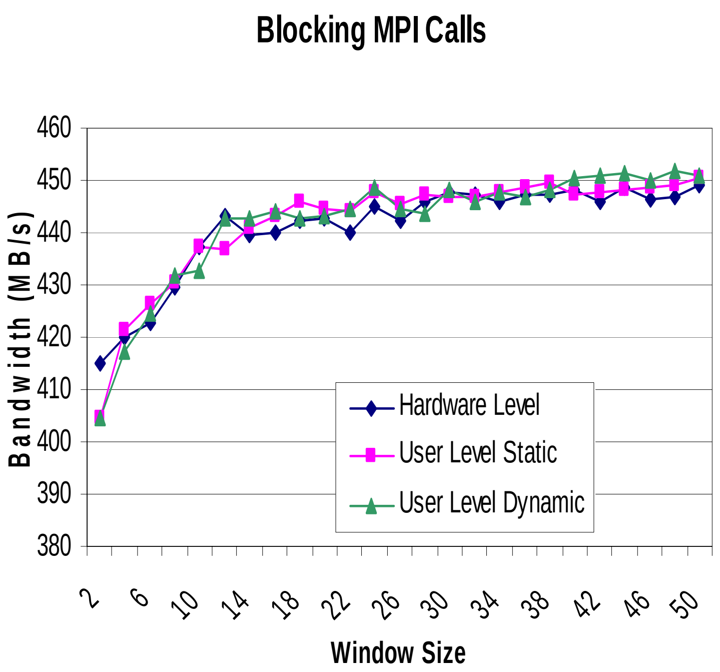
- With enough buffers, all schemes perform comparably for small messages
- Blocking and Non-blocking MPI calls performs comparably for small messages because message are copied and sent eagerly

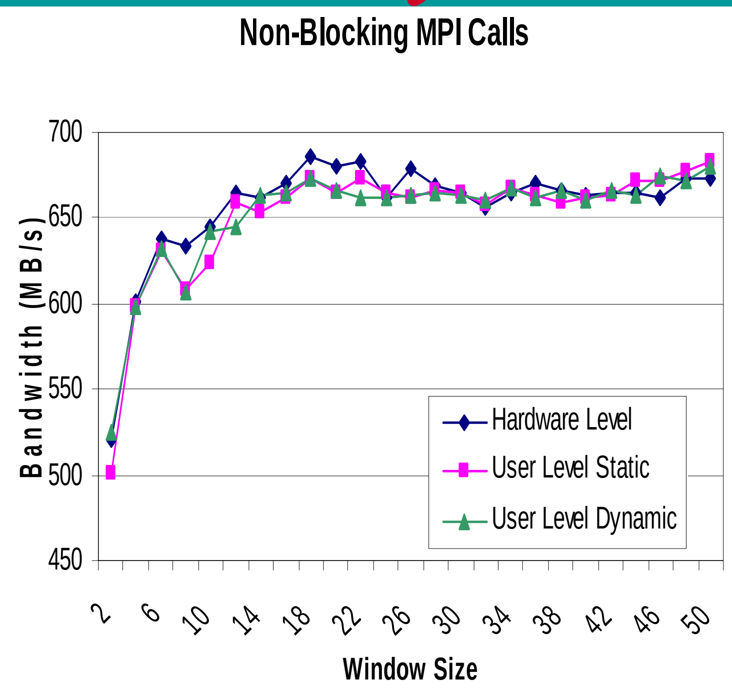
MPI Bandwidth (Prepost = 10 and Size = 4 Bytes)



- Buffers are not enough, which triggers flow control mechanisms
- user level dynamic performs the best, user level static performs the worst

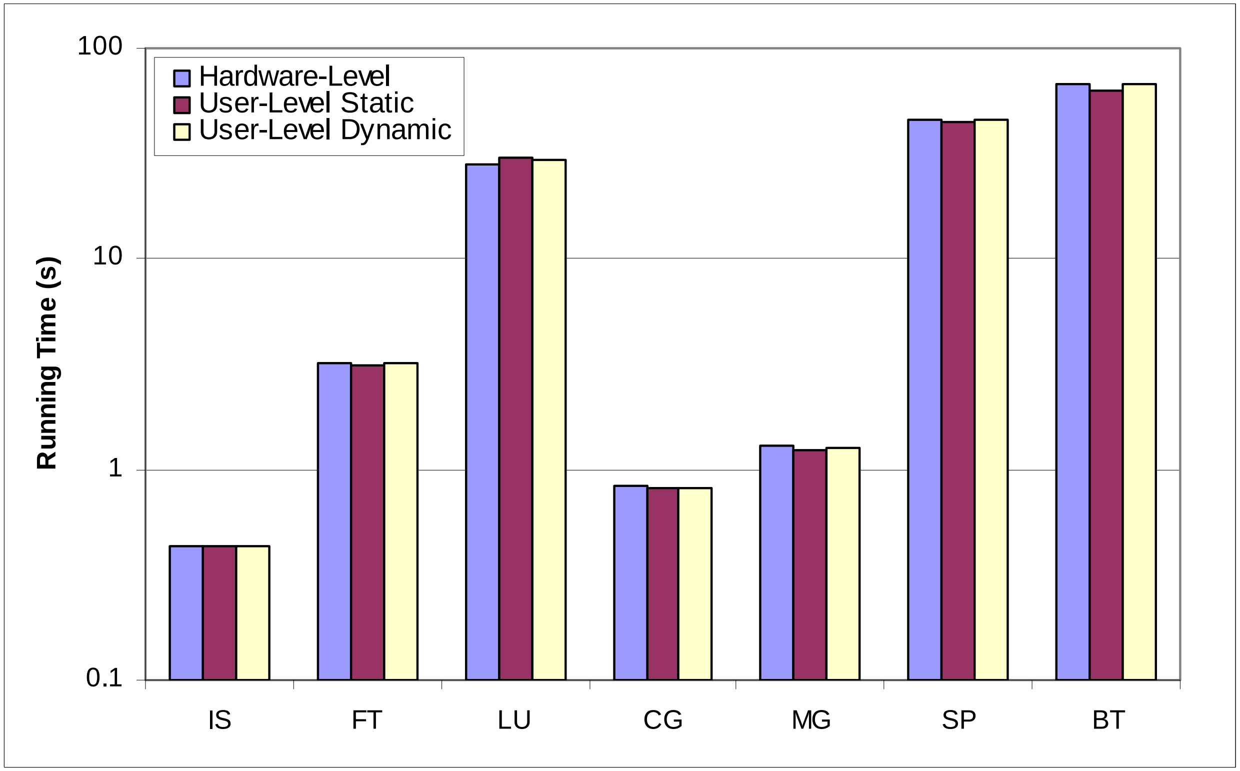
MPI Bandwidth (Prepost = 10 and Size = 32 KB)





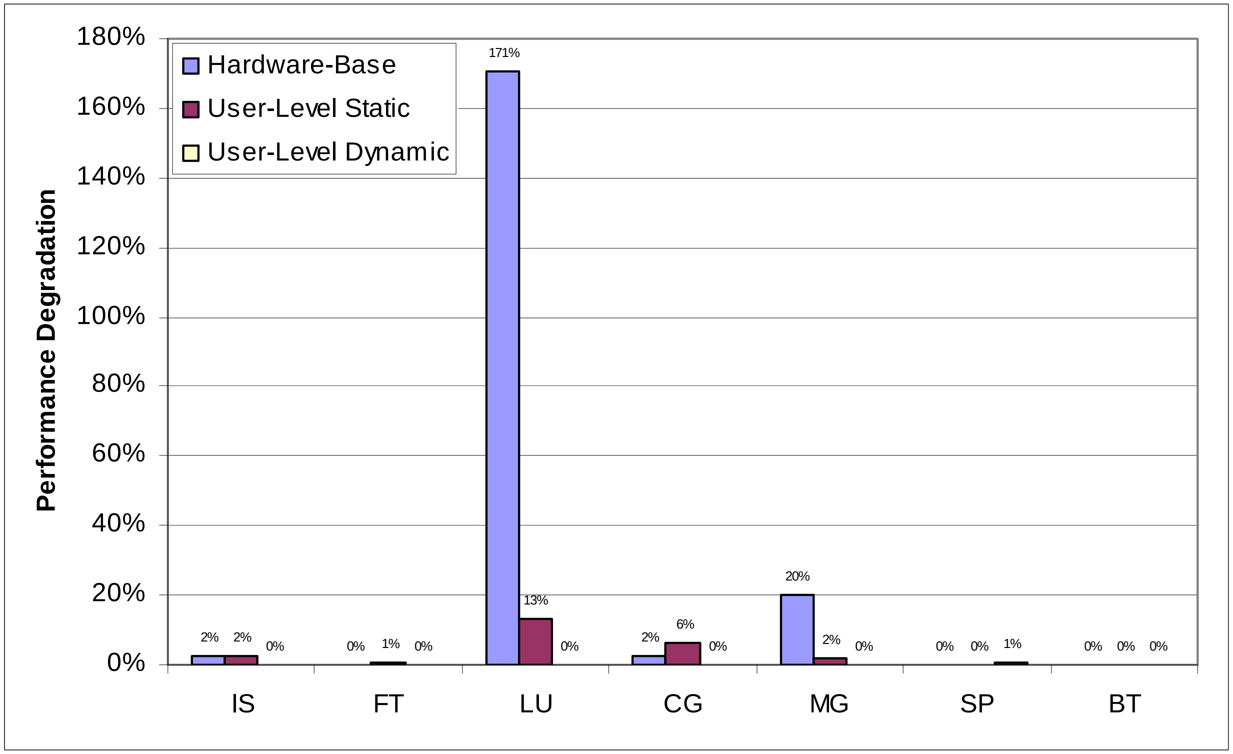
- Large messages use Rendezvous protocol which has two-way traffic
- All schemes perform comparably
- Non-blocking calls give better performance

NAS Benchmarks (Pre-post = 100)



- ➢All schemes perform comparably when given enough buffers

NAS Benchmarks (Pre-post = 1)



- ➢Even with very few buffers, most applications still perform well
- ➢LU performs significant worse for hardware based and user-level static
- ➢Overall, user-level dynamic gives best performance

Explicit Credit Messages for User-Level Static Schemes

App	#ECM	#Total Msg
IS	0	383
FT	0	193
LU	9002	48805
CG	0	4202
MG	1	1595
BT	0	28913
SP	0	14531

- Piggybacking is quite effective
- In LU, the number of explicit credit messages is high

Maximum Number of Buffers for User-Level Dynamic Schemes

App	#Buffer
IS	4
FT	4
LU	63
CG	3
MG	6
BT	7
SP	7

- Almost all applications only need a few (less than 8) buffers per connection for optimal performance
- LU requires more buffers

Conclusions

- Three different flow control schemes for MPI over InfiniBand
- Evaluation in terms of overhead and buffer efficiency
- Many applications (like those in NAS) require a small number of buffers for each connection
- User-Level Dynamic Scheme can achieve both good performance and buffer efficiency

Future Work

- More application level evaluation
- Evaluate using larger scale systems
- Integrate the schemes with our RDMA based design for small messages
- Exploit the recently proposed Shared Receive Queue (SRQ) feature

Web Pointers

NBC home page

http://www.cis.ohio-state.edu/~panda/
http://nowlab.cis.ohio-state.edu/

MVAPICH home page

http://nowlab.cis.ohio-state.edu/projects/mpi-iba/